



A Machine Learning Approach for Game Bot Detection Through Behavioural Features

Mario Luca Bernardi¹, Marta Cimitile^{2(✉)}, Fabio Martinelli³,
and Francesco Mercaldo³

¹ Giustino Fortunato University, Benevento, Italy

² Unitelma Sapienza, Roma, Italy

`marta.cimitile@unitelma.it`

³ Institute for Informatics and Telematics,
National Research Council of Italy (CNR), Pisa, Italy

Abstract. In the last years, online games market has been interested by a sudden growth due to the birth of new gaming infrastructures that offer more effective and innovative services and products. Simultaneously to the diffusion of on line games, there was an increasing use of game bots to automatically perform malicious tasks. Game bots users aim to obtain some rewards by automating the most tedious and prolonged activities arousing the disappointment of the game community. Therefore, the detection and the expulsion of game bots from the game environment, become critical issues for the game's developers that want to ensure the satisfaction of all the players. This paper describes an approach for the game bot detection in the online role player games consisting to distinguish between game bots and human behavior and based on the adoption of supervised and unsupervised machine learning techniques. These techniques are used to discriminate between users and game bots basing on some user behavioral features. The approach is applied to a real-world dataset of a popular role player game and the obtained results are encouraging.

Keywords: Game bot · Machine learning · Cluster analysis
Game bot detection · Security · Testing

1 Introduction

The worldwide games market is largely grown in the last years (the revenue increases of 7.8% in the first quarter of 2017 with respect to the year before)¹ due to the development of innovative online gaming platforms and the availability of a huge and high-quality games offering. As a matter of fact, an increasing interest

¹ <https://newzoo.com/solutions/standard/market-forecasts/global-games-market-report/>.

has been pointed to the on line games, that thanks to the diffusion of Internet network, are become very popular. Differently from traditional games, on line games are partially or primarily played through the Internet network or another computer network [1]. These games are made available through modern gaming platforms that include PCs, consoles and mobile devices, and offer many genres (i.e., first-person shooters, strategy games and massively multiplayer online role-playing games) [2]. Moreover, online games can have a different design (from simple text-based environments to more complex graphics and virtual worlds) [3] and attract players from a variety of ages, nationalities, and occupations [4, 5]. However, given their diffusion and their characteristics, on line games represent an appealing scenario for attackers to perpetrate illegal activities in the online world [6]. They are interested to perform illegal activities for several reasons: get economic gain (cyber assets can be changed into real currency) [7], capture reserved information about the other players or became popular in the game community. Several malicious tasks are performed by using game bots able to repeat continuously a given activity in the online game. Game bots consist of an artificial intelligent software that simulates human behavior in a video game [8]. Since they automatize playing actions, game bots can earn more cyber assets than human users because they can play at a full time without any break. Moreover, game bots cause harm to the human users because they consume game resources reducing the player opportunities: for example, game bots defeat all monsters and harvest the available items before the human users. Basing on the above considerations, we can conclude that game bots damage the game provider reputation and reduce the game's lifecycle [9]. This paper extends and explores a method for game bots detection in a Multiplayer Online Role-Playing Game (MMORPG environment) proposed in [10, 11]. MMORPGs are a kind of online game where a large number of players interact with one another within a virtual world. The method uses some features (i.e., Player Information, Player Actions, Group Activities, Social Interaction Diversity and Network Measures) to distinguish game bots from human players behavior. This discrimination is performed by using some classifiers built thought supervised and unsupervised (i.e., cluster analysis) machine learning techniques. The effectiveness of our proposed approach is evaluated in a real context consisting of a dataset obtained from the operation of Aion: The Tower of Eternity² game. It is a MMORPG fantasy free-to-play popular game where all the operations performed by real players and game bots are identified and labeled by the game company. The rest of the paper is organized as follows: the next section provides an overview of the related work. In Sects. 3 and 4 the proposed features and the detection techniques are illustrated. In Sect. 5, authors evaluate if it is possible to detect game bots by using the set of the proposed behavioral features and evaluate the effectiveness of the proposed approach to discriminate human and botnet behaviors in a real MMORPG game. Section 5 shows the results of the evaluation finally, conclusion and future works are given in the last section.

² <https://en.aion.gameforge.com/website/>.

2 Related Work

A great number of approaches are recently proposed on the topic of game bots detection. Several methods adopt data mining techniques to analyze the log data extracted by the game server. These approaches are favorite by the game service providers because they allow to detect and block the game bots without any interference with the game user playing and without the deployment of additional software on the client-side. Some server-side approaches are focused on the analysis of users interactions and social activity [12] assuming that game bots and human player have different game style discriminating them (these approaches are also called behavioral approaches). For example, in [13], the social behaviors of both human and game bots are modeled and compared through the use of behavioral features. In [14], some thresholds are fixed to discriminate the different behaviors of the game bots and the human players basing on their activities. In [15], the bot detection is performed by analyzing the sequences of window events (described as a set of attributes) produced by the game players. Learning algorithms are used to distinguish and classify human from game bots event sequences. Authors, in [16, 17] describe a manifold learning approach consisting to analyze the avatars movement trajectories to distinguish game bots from human players (they are assumed to have different movement trajectories). Moreover, in [18] authors compare the traffic generated by the game bots versus human players to perform game bots detection. A specific focus on the MMORPG environment is in [19] where the characteristics of a MMORPG environment are analyzed and a method for detecting GFGs (i.e., gold farming groups) is explored. MMOPG is also considered in [20]. Here, authors perform action sequence analysis on a server-side account theft detection system to protect game users from malicious players. Moreover, several approaches are based on the analysis of behavioral approaches [9, 21–24]. For example, [13] uses some features to capture the social behavior of the game players assuming that game bots and humans tend to form their social network in different ways. The behavioral approaches analyze a limited number of features (one or two behavioral features) having, as consequence, a high dependence of the approach from the single game domain. This limit is discussed and faced in [25], where authors propose a new approach based on a larger set of features. It uses both game and player-related features similar to our proposed approach. With respect to [25], in our proposed approach the obtained results show a higher precision in the game bot detection. Moreover, differently from [25], in this work, a feature selection step is provided in order to improve the performance in the real environments. This step aims to select the most discriminative features (they will become input for the classification step) between human users and game bot and the time employed to build the classifiers is evaluated as the maximum time interval to detect a game bot behavior.

3 Background

In order to evaluate the effectiveness of the behavioral feature sets to discriminate between human and game bot, we use supervised and the unsupervised [26] machine learning approaches. Indeed, machine learning tasks are typically classified into these two categories, depending on the nature of the learning available to a learning system [27, 28]. Supervised learning is the machine learning task of inferring a function from the labeled training data. In this case, the training data consist of a set of training examples. In supervised learning, each example is a pair consisting of an input object (i.e., the behavioral feature sets) and the desired output value (i.e., the game bot of the human label). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new features without a label. The optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. Differently, the unsupervised machine learning (also called cluster analysis) is the machine learning task of inferring a function to describe hidden structure from “unlabeled” data (a classification or categorization is not included in the observations). For this reason, using this approach is possible to infer the human and the game bot label without previous knowledge about the nature of the feature sets considered (i.e., these algorithms do not require the label). Relating to supervised machine learning classification, we consider six different algorithms of classification: J48, DecisionStump, HoeffdingTree, RandomForest, RandomTree and REPTree [29, 30]. The supervised used algorithms are listed in the following:

- *J48* [31]: It is an open source Java implementation of the C4.5 decision tree algorithm. It is a statistical classifier based on id3 algorithm [32]. Basically, it computes the potential information for every considered attribute, given by a test on the attribute and the gain in information is calculated that would result from a test on the attribute.
- *DecisionStump* [33]: A decision tree-based model with one internal root node immediately connected to the terminal nodes. The prediction is allowed by a decision stump basing on the value of just one input feature.
- *HoeffdingTree* [34]: An incremental, anytime decision tree induction algorithm learning from massive data streams. Basing on the assumption that the distribution generating examples does not change over time, it often uses a small sample to choose an optimal splitting attribute.
- *RandomForest* [35]: It operates by constructing, at training time, a multitude of decision trees and outputting the class that is the mode (the most frequent value appearing in a set of data) of the classes of the individual trees.
- *RandomTree* [36]: It constructs a tree containing some randomly chosen attributes at each node. No pruning is performed.
- *REPTree* [37]: It builds a decision tree using information gain/variance and reduces errors using reduced-error pruning. The values are ordered for numeric attributes once and missing values are recovered with by splitting the corresponding instances into pieces.

With regards to the unsupervised machine learning classification, we consider the following six cluster analysis algorithms: SimpleKMeans, FarthestFirst, FilteredClusterer, MakeDensityBasedClusterer, Cobweb, DBScan.

- *SimpleKMeans* [38]: Cluster data using the k-means algorithm with the Euclidean distance. The k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.
- *FarthestFirst* [39]: The farthest-first traversal of a bounded metric space computes a sequence of points in the space, where the first point is selected arbitrarily and each successive point is as far as possible from the set of previously-selected points.
- *FilteredClusterer* [40]: Algorithm for running an arbitrary clusterer on data that has been passed through an arbitrary filter. Like the clusterer, the structure of the filter is based exclusively on the training data and test instances will be processed by the filter without changing their structure.
- *MakeDensityBasedClusterer* [41]: Algorithm for wrapping a clusterer to return a distribution and density. It fits normal distributions and discrete distributions within each cluster produced by the wrapped clusterer.
- *Cobweb* [42]: It is an incremental system for hierarchical conceptual clustering. This algorithm incrementally organizes observations into a classification tree. Each node in a classification tree represents a class (concept) and it is labeled by a probabilistic concept that summarizes the attribute-value distributions of objects classified under the node. This classification tree can be used to predict missing attributes or the class of a new object.
- *DBScan* [43]: The Density-based spatial clustering of applications with noise (DBSCAN) is a data clustering algorithm. It is a density-based clustering algorithm: given a set of points in some space, it groups together points that are closely packed together (points with many nearby neighbors), marking as outliers the points that lie alone in low-density regions (whose nearest neighbors are too far away). DBSCAN is one of the most common clustering algorithms and also most cited in scientific literature.

4 The Method

As discussed in Sect. 1, our proposed approach aims to discriminate game bots from human players basing on their different behavior in the context of MMORPG games. For this reason, we propose a set of behavioral features and we evaluate their capability to discriminate human users and game bots. The considered features were extracted and computed by a real game company (as described in [9]). They are grouped in the following categories: Player Information (PI), Player Actions (PA) (they describe the player’s characteristics), Group Activities (GA), Social Interaction Diversity (SID) and Network Measures (NM). The first two features groups describe players characteristics, while the other features categories are referred to the game characteristics. Table 1 shows the considered features for each category listed above.

Table 1. The features involved in the study with the correspondent category [44].

Category	Features
Player Information	PI ₁ , PI ₂ , PI ₃ , PI ₄
Player Actions	PA ₁ , PA ₂ , PA ₃ , PA ₄ , PA ₅ , PA ₆ , PA ₇ , PA ₈ , PA ₉ , PA ₁₀
Group Activities	GA ₁ , GA ₂
Social Interaction Diversity	SID ₁
Network Measures	NM ₁ , NM ₂ , NM ₃ , NM ₄ , NM ₅ , NM ₆ , NM ₇ , NM ₈ , NM ₉

Looking to the table, the Player Information (PI) features describe how the players perform some game tasks. The main assumption is that the PI features analysis shows a gap between the values of these features for game bots and for human users. The considered PI features are the followings: login frequency (PI₁), play time (PI₂), game money (PI₃) and a number of IP address (PI₄). Similarly, the PA features are introduced to investigate whether the player actions can reflect the differences between game bots and human users. For instance, game bots are often connected to the game at a full time and can play for 24 consecutive hours, differently from human users, that typically are not able to play during several time-windows (i.e., work time, sleeping time). This allows them to reach a certain rank level and obtain more powers to fight the enemies effectively. Moreover, more player kill points can be acquired by defeating players of opposing factions. Player kill points can be used to purchase various items from vendors and to determine a player's rank within the game world. In the Aion game³, the more player kill points a player has, the higher is the rank of the player. The high ranking player can feel a sense of accomplishment. The attackers are interested to obtain game powers because they can resell these powers to the other game users⁴. Finally, the game bots sit more frequently than human users in order to recover health and many points. The considered PA features are: sitting (PA₁) (i.e., an action taken by players to recover their health), earning experience points (PA₂), obtaining items (PA₃), earning game money (PA₄), earning player kill points (PA₅), harvesting items (PA₆), resurrecting (PA₇), restoring experience points (PA₈), being killed by a non-player and/or player character (PA₉), and using portals (PA₁₀). The GA features are introduced to evaluate the different behavior of game bots and human players with respect to their participation in the group activities. The rationale behind the GA feature is that there is a gap between the values of the social features of game bots and those of human users because game bots do not attempt to social as humans. For example, game bots are not interested in the activities that allow increasing the socialization among the player's community but they only perform social activities that are required to obtain game advantages. As matter of fact, as a consequence of the game bot protract play time, we expected

³ <https://sites.google.com/a/hksecurity.net/ocslab/Datasets/game-bot-detection>.

⁴ <http://www.geek.com/games/>.

that the GA category is different between game bots and human users. The GA category includes the average duration of party play (GA_1) and the number of guild activities (GA_2). For example, looking to the GA_1 , party play is a group play formed by two or more players in order to undertake quests or missions together. Party play aims to complete difficult quests by collaboration and enjoy socialization. Differently, from other players, game bots perform party play, with the only goal to acquire game money and items faster and more efficiently in order to obtain powers. For this reason, we expected that the value obtained for GA_1 is different between game bots and human users. The concept of party play is also useful to understand the SID_1 features defined as the entropy of party play. Game bots concentrate only on particular actions, whereas human users execute multiple tasks as needed to thrive in the online game world. Finally, the NM category regards the player's social interaction network that can be represented as a graph with characters as the nodes and interactions between them as the edges. An edge between two nodes (players) in this graph may, for example, highlight the transfer of an item between the two nodes. The features of NM include the degree centrality (NM_1), betweenness centrality (NM_2), closeness (NM_3), eigenvector centrality (NM_4), eccentricity (NM_5), authority (NM_6), hub (NM_7), PageRank (NM_8), and clustering coefficient (NM_9). The NM category features are explained in Table 2.

5 The Evaluation

The proposed experiment aims to evaluate the effectiveness of the feature vector we propose, with respect to the goals stated in the introduction. More specifically, the capability of the behavioral features to discriminate the game bots by the human user behavior is here evaluated. A classification of the proposed features values is carried out by using several state-of-the-art machine learning classifiers built with the behavioral feature categories we considered. In this study, a real dataset, obtained from the operation of a popular and free available game called Aion, is used. It contains all in-game action logs for 88 days, between April 9th and July 5th of 2010. During this period, 49,739 players attend the game for at least three hours. Among these players, 7702 characters are identified by the game company as game bots. The banned list provided by the game company is used in our evaluation as the ground truth after that each banned user has been vetted and manually verified by human labor and active monitoring. Starting by the log released by the game company, in our dataset we aggregated the values of the features related to the same user and we labeled the feature vectors as "human players" or "game bot". Moreover, in the dataset all the private and personal information of the players are protected by the anonymity. The evaluation consists of some main steps: (i) a comparison of the descriptive statistics of the behavioral features populations; (ii) the hypotheses testing, aiming to verify if the feature categories have different distributions for the populations of game bot and human behavior; and (iii) the classification analysis aimed at assessing whether the behavioral feature categories are able to correctly classify game

Table 2. The features belonging to the Network Measures category with their description [44].

NM category feature	Description
Degree centrality	This features represents the centrality focused on the degree. The more edges an actor has, the more important it is
Betweenness centrality	It counts the number of shortest paths between two nodes on which a given actor resides
Closeness centrality	An actor is considered important if it is relatively close to all other actors. Closeness is based on the inverse of the distance of each actor to every other actor in the network
Eigenvector centrality	Indicates that a given node has a relationship with other valuable nodes. A high eigenvector value for an actor means that a node has several neighbors with high eigenvector values
Eccentricity	The eccentricity of node v is calculated by computing the shortest path between node v and all other nodes in the graph; then the longest shortest path is chosen
Authority	Exhibits a node pointed to by many good hubs
Hub	Exhibits a node that points to many good authorities
PageRank	Assigns a numerical weight to each element of a hyperlinked set of documents, such as the World Wide Web, with the purpose of “measuring” its relative importance within the set
Clustering coefficient	It quantifies how close neighbors are to being a clique: a clique is a subset of all of the edges connecting pairs of vertices of an undirected graph

bot and human behavior. The classification analysis is performed by using both supervised and unsupervised machine learning algorithms.

5.1 Descriptive Statistics

In this section the box plots of the distribution of game bot and human behavior features are shown in order to demonstrate their differences. Figures 1, 2 and 3 show the box plots for a subset of features belonging to each category considered in the study. Figure 1(a) shows the box plot related to the login frequency time feature (PI_1) between game bot and human users. Since they assume similar values, we can conclude that game bot and human user login with similar frequency. In our opinion, the limit of this feature is that it does not consider the login time but only the login frequency. Figure 1(b) shows the box plot for the play time feature (PI_2). The figure highlights that the distributions between game bots and human users are very different: as matter of fact, while human user presents a small distribution, the game bot one exhibits higher play time

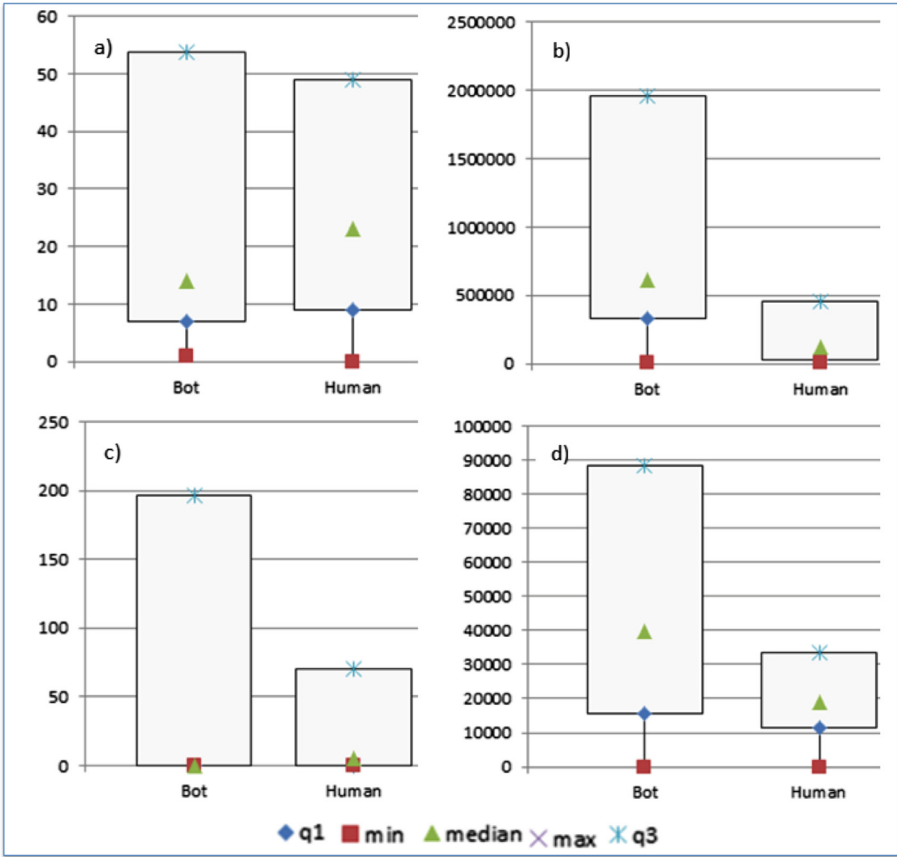


Fig. 1. The box plot relating to the game bot and human distributions for the login frequency feature (a) the play time feature (b) the sitting feature (c) and for the earning experience points feature (d).

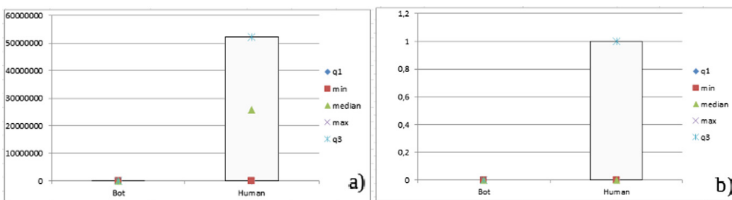


Fig. 2. The box plot relating to the game bot and human distributions for the party play time feature (a) and the guild activities feature (b), belonging to the Group Activities category.

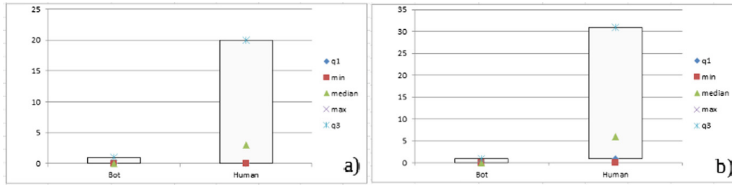


Fig. 3. The box plot relating to the game bot and human distributions for the degree centrality feature (a) and the betweenness centrality feature (b), belonging to the Network Measures category.

values. The reason is that game bot is able to play the game without any break, differently from human users.

Figure 1(c) and (d) show the box plot related to game bot and human distributions for the feature belonging to the Player Actions category. Figure 1(c) shows that the sitting time distribution for game bots seems to be wider if compared with the human users one. This feature is related to the number of rounds that game bots and human users are able to play.

Figure 1(d) shows the box plot related to earning experience points feature (PA_2). This feature evaluates the ability to earn points in order to buy power for the character. The figure shows a wider dimension of game bots distribution with respect to the human users one confirming that the game bots are able to gather more points if compared with human users.

Figure 2(a) shows the box plot related to the party play time feature (GA_1). The distribution obtained for the human user box plots is wider if compared with the game bots one. This happens because game bots usually have not interest to make party play in order to play or socialize with other users: the focus of game bots is only to disturb human users and gather points in order to have character reinforcements.

Figure 2(b) shows the box plot related to the guild activities feature (GA_2). The feature is related to the capacity of a player to perform missions in cooperation with other players. Confirming the previous box plot, game bot has no interest to cooperate in the play, while human user usually needs to play together in order to complete successfully difficult missions.

Figure 4 shows the distributions for the SID_1 feature. The distributions appear very similar, highlighting that both game bots and human users interact with them, for this reason, the considered feature is not discriminative between the observed populations.

Figure 3(a) shows the box plot related to the NM_1 feature. These box plots exhibit that human users present a wider degree centrality if compared with game bots. We conclude that human users have more friends if compared with game bots.

Figure 3(b) shows the NM_2 feature box plot. As in the previous box plots, the human user's distribution is wider than the game bots one. It happens because the human user tries to reach an objective by using the shortest path, differently from game bots that do not consider this.

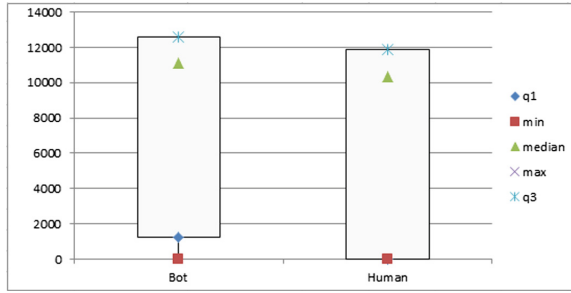


Fig. 4. The box plot relating to the game bot and human distributions for the party play entropy, belonging to the Social Interaction category [44].

5.2 Hypothesis Testing

Basing on the research goals discussed in Sect. 1, the hypotheses testing and the null hypothesis can be the following:

H_1 : There are statistically significant differences between the considered features of game bots and human users

H_0 : There are no statistically significant differences between the considered features of game bots and human users

The null hypothesis was tested with three different tests in order to enforce the conclusion validity: Mann-Whitney, Kolmogorov-Smirnov Test and Wald-Wolfowitz. For all the tests the p-level is fixed to 0.05. The purpose of these tests is to determine the level of significance, i.e., the risk (the probability) that erroneous conclusions be drawn: since we set the significance level equal to 0.05, we accept to make mistakes 5 times out of 100. The hypothesis testing aims at evaluating if the features present different distributions for the populations of the game bot and human behavioral characteristics with statistical evidence. We assume valid the results when the null hypothesis is rejected by both the tests performed. Table 3 shows the results of hypothesis testing: the null hypothesis H_0 can be rejected for all the features. This means that there is statistical evidence that the feature vector is a potential candidate for correctly classifying between game bot and human behavioral characteristics. This result will provide an evaluation of the risk enforced by the fact that three tests are used and the selected features produce values which belong to two different distributions (i.e., the one related to the game bots and the human users). With the classification analysis, we will be able to establish the accuracy of the features in associating any behavioral feature to the game bot or to the human distribution.

5.3 Classification Analysis

This step consists of building classifiers in order to evaluate the feature vector accuracy to distinguish between game bots and human users. Four metrics were

Table 3. Results of the hypothesis test of the null hypothesis H_0 .

Variable	Mann-Whitney	Kolmogorov-Smirnov	Wald-Wolfowitz
PI1	0.000000	p < .001	0.000
PI2	0.000000	p < .001	0.000
PI3	0.000000	p < .001	0.000
PI4	0.000000	p < .001	0.000
PA1	0.000000	p < .001	0.000
PA2	0.000000	p < .001	0.000
PA3	0.000000	p < .001	0.000
PA4	0.000000	p < .001	0.000
PA5	0.000000	p < .001	0.000
PA6	0.000000	p < .001	0.000
PA7	0.000000	p < .001	0.000
PA8	0.000000	p < .001	0.000
PA9	0.000000	p < .001	0.000
PA10	0.000000	p < .001	0.000
GA1	0.000000	p < .001	0.000
GA2	0.000000	p < .001	0.000
SI1	0.000000	p < .001	0.000
NM1	0.000000	p < .001	0.000
NM2	0.000000	p < .001	0.000
NM3	0.000000	p < .001	0.000
NM4	0.000000	p < .001	0.000
NM5	0.000000	p < .001	0.000
NM6	0.000000	p < .001	0.000
NM7	0.000000	p < .001	0.000
NM8	0.000000	p < .001	0.000
NM9	0.000000	p < .001	0.000

used to evaluate the classification results: Precision, Recall, F-Measure and ROC Area. The precision has been computed as the proportion of the examples that truly belong to class X among all those which were assigned to the class. It is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved:

$$Precision = \frac{tp}{tp + fp}$$

where tp indicates the number of true positives and fp indicates the number of false positives. The recall has been computed as the proportion of examples that were assigned to class X, among all the examples that truly belong to the class,

i.e., how much part of the class was captured. It is the ratio of the number of relevant records retrieved to the total number of relevant records:

$$Recall = \frac{tp}{tp + fn}$$

where tp indicates the number of true positives and fn indicates the number of false negatives. The F-Measure is a measure of a test's accuracy. This score can be interpreted as a weighted average of the precision and recall:

$$F\text{-Measure} = 2 * \frac{Precision * Recall}{Precision + Recall}$$

The Roc Area is defined as the probability that a positive instance randomly chosen is classified above a negative randomly chosen. The classifier training is performed by defining T as a set of labeled traces (M, l) , where each M is associated to a label $l \in \{B, H\}$ (where B represents the game bot, while H the human user). For each M , a feature vector $F \in R_y$ is build, where y is the number of the features used in training phase ($y = 4$ for the PI category, $y = 9$ for the PA category, $y = 2$ for the GA category, $y = 1$ for the SID category and $y = 9$ for the NM category). In the learning phase, we use a k -fold cross-validation: the dataset is randomly partitioned into k subsets. A single subset is retained as the validation dataset for testing the model, while the remaining $k - 1$ subsets of the original dataset are used as training data. We repeated the process for $k = 10$ times; each one of the k subsets has been used once as the validation dataset. To obtain a single estimate, we computed the average of the k results from the folds. We evaluated the effectiveness of the classification method with the following procedure:

1. build a training set $T \subset D$;
2. build a testing set $T' = D \div T$;
3. run the training phase on T ;
4. apply the learned classifier to each element of T' .

Each classification was performed using 20% of the dataset as training dataset and 80% as testing dataset employing the full feature set. The obtained results are shown in Table 4.

Looking at the table, we can conclude that a precision equal to 0.952 and a recall of 0.953 is obtained by using the J48 algorithm to classify the feature belonging to PI category. Similarly, a precision equal to 0.954 and a recall of 0.955 is obtained with the RandomForest algorithm classifying the feature belonging to PA category while RepTree algorithm shows a precision equal to 0.858 and a recall a 0.882 on the feature belonging to GA category. J48 also gives a precision equal to 0.836 and a recall equal to 0.875 on the feature belonging to SID category. Finally, a precision equal to 0.923 and a recall 0.928 using the RandomForest algorithm classifying the feature belonging to NM category. The table also shows that we obtained the best results (in terms of precision and recall) when classifying the features related to the PI and PA categories.

Table 4. Classification results: Precision, Recall, F-Measure and RocArea for classifying the feature categories, computed with six different classification algorithms [44].

Category	Algorithm	Precision	Recall	F-Measure	Roc Area	Time
<i>PI</i>	J48	0.95	0.951	0.949	0.856	1.97
	DecisionStump	0.942	0.944	0.941	0.823	0.21
	HoeffdingTree	0.941	0.944	0.941	0.872	0.3
	RandomForest	0.952	0.953	0.950	0.895	37.8
	RandomTree	0.917	0.916	0.916	0.814	0.64
	REPTree	0.946	0.948	0.945	0.880	0.93
<i>PA</i>	J48	0.948	0.950	0.947	0.862	7.01
	DecisionStump	0.934	0.936	0.935	0.830	0.58
	HoeffdingTree	0.942	0.944	0.942	0.872	0.94
	RandomForest	0.954	0.955	0.953	0.95	57.23
	RandomTree	0.952	0.952	0.921	0.901	4.01
	REPTree	0.948	0.950	0.948	0.888	3.36
<i>GA</i>	J48	0.858	0.881	0.843	0.764	0.38
	DecisionStump	0.764	0.874	0.816	0.721	0.05
	HoeffdingTree	0.854	0.880	0.839	0.783	0.17
	RandomForest	0.820	0.855	0.833	0.766	17.71
	RandomTree	0.820	0.854	0.833	0.758	0.42
	REPTree	0.858	0.882	0.845	0.784	0.47
<i>SID</i>	J48	0.836	0.875	0.821	0.698	0.37
	DecisionStump	0.764	0.874	0.816	0.690	0.06
	HoeffdingTree	0.826	0.874	0.823	0.710	0.32
	RandomForest	0.805	0.863	0.822	0.688	8.31
	RandomTree	0.805	0.865	0.822	0.682	0.14
	REPTree	0.829	0.874	0.823	0.717	0.11
<i>NM</i>	J48	0.917	0.923	0.917	0.810	24.75
	DecisionStump	0.871	0.884	0.875	0.673	0.99
	HoeffdingTree	0.892	0.903	0.891	0.769	2.68
	RandomForest	0.923	0.928	0.923	0.858	42.42
	RandomTree	0.886	0.887	0.887	0.751	0.69
	REPTree	0.917	0.923	0.917	0.845	5.5

Table 5 shows the results with the unsupervised machine learning classification.

We obtained the best precision result by using the MakeDensityBasedClusterer algorithm (i.e., 0.954) and the best recall is obtained by using the same classification algorithm (0.955) in the PA category feature classification. We

Table 5. Classification results: Precision, Recall, F-Measure and RocArea for classifying the feature categories, computed with six different unsupervised classification algorithms.

Category	Algorithm	Precision	Recall	F-Measure	Roc Area	Time
<i>PI</i>	SimpleKMeans	0.93	0.93	0.93	0.843	1.79
	FarthestFirst	0.92	0.93	0.92	0.839	1.98
	FilteredClusterer	0.91	0.92	0.907	0.841	1.89
	MakeDensityBasedClusterer	0.905	0.918	0.911	0.840	49.06
	Cobweb	0.927	0.919	0.922	0.839	39.58
	DBScan	0.934	0.948	0.971	0.836	42.73
<i>PA</i>	SimpleKMeans	0.948	0.927	0.964	0.871	2.37
	FarthestFirst	0.939	0.935	0.936	0.872	3.84
	FilteredClusterer	0.926	0.934	0.92	0.868	3.94
	MakeDensityBasedClusterer	0.954	0.955	0.953	0.845	47.85
	Cobweb	0.926	0.918	0.921	0.862	44.25
	DBScan	0.921	0.924	0.922	0.873	40.68
<i>GA</i>	SimpleKMeans	0.839	0.874	0.856	0.722	1.92
	FarthestFirst	0.801	0.827	0.813	0.747	3.29
	FilteredClusterer	0.833	0.879	0.855	0.779	3.98
	MakeDensityBasedClusterer	0.838	0.875	0.856	0.739	31.48
	Cobweb	0.811	0.825	0.817	0.749	32.19
	DBScan	0.814	0.830	0.821	0.751	35.28
<i>SID</i>	SimpleKMeans	0.827	0.841	0.833	0.719	2.01
	FarthestFirst	0.787	0.798	0.792	0.702	1.93
	FilteredClusterer	0.799	0.778	0.788	0.705	2.16
	MakeDensityBasedClusterer	0.799	0.801	0.799	0.718	37.98
	Cobweb	0.809	0.817	0.812	0.702	41.85
	DBScan	0.789	0.799	0.799	0.793	39.84
<i>NM</i>	SimpleKMeans	0.902	0.919	0.914	0.819	3.06
	FarthestFirst	0.899	0.857	0.877	0.709	2.87
	FilteredClusterer	0.899	0.854	0.876	0.714	2.48
	MakeDensityBasedClusterer	0.923	0.928	0.923	0.858	42.42
	Cobweb	0.876	0.890	0.882	0.744	43.87
	DBScan	0.839	0.875	0.853	0.732	44.28

highlight in addition that with regards to the PA category using the other algorithms the worst precision and recall values are obtained by the DBScan algorithm (0.921 of precision and 0.924 of recall). With the classification of the PI features category, we obtain a precision ranging from 0.91 (with the FilteredClusterer algorithm) to 0.934 (with the DBScan algorithm) and a recall ranging

from 0.918 (using the `MakeDensityBasedClusterer`) to 0.948 with the `DBSCAN` algorithm. For the others features categories we obtained the following results:

- with regards to the GA category, a precision ranging from 0.801 (with the `FarthestFirst` algorithm) to 0.839 (with the `SimpleKMeans` algorithm) and a recall ranging to 0.825 (with the `Coweb` algorithm) to 0.879 (with the `FilteredClusterer` algorithm);
- with regards to the SID category, a precision ranging from 0.787 (with the `FarthestFirst` algorithm) to 0.809 (with the `Coweb` algorithm) and a recall ranging to 0.778 (with the `FilteredClusterer` algorithm) to 0.841 (with the `SimpleKMeans` algorithm);
- with regards to the NM category, a precision ranging from 0.839 (with the `DBSCAN` algorithm) to 0.923 (with the `MakeDensityBasedClusterer` algorithm) and a recall ranging to 0.854 (with the `FilteredClusterer` algorithm) to 0.928 (with the `MakeDensityBasedClusterer` algorithm).

Looking to the tables, the best feature categories, in both the supervised and the unsupervised classifications, are the PA and PI category; for this reason, we perform a feature selection on these categories in order to investigate whether using a small feature set we are able to obtain better results. As matter of fact, the feature selection is employed to improve the ability of classifier in discriminating human and game bot instances and decrease training time. The used feature selection algorithm is the `BestFirst`, that implements a best-first search strategy to navigate attribute subsets which basically explores a graph by expanding the most promising node chosen according to a specified rule. We obtained that the most discriminating feature in the PI category is just the play time (PI_2), while in PA category there are four best features: sitting (PA_1), earning experience points (PA_2), obtaining items (PA_3) and earning player kill points (PA_5). In order to evaluate if the new features set belonging to PI and PA categories are able to overcome the results obtained in Table 4, we build different (supervised and unsupervised) classifiers. Table 6 shows the results we obtained using the supervised classification algorithms, while Table 7 shows the results we obtained using the unsupervised classification algorithms.

The values of precision and recall are increased for both the features categories we considered. For the PI category, the best precision value (`RandomForest`) is incremented from 0.952 to 0.954 and the recall one is incremented from 0.953 to 0.984. For the PA category, the best precision value (`RandomForest`) it is incremented from 0.954 to 0.960 and the recall one is incremented from 0.955 to 0.986. The time analysis confirms the effectiveness of the feature selection step in order to quickly identify the game bot: as matter of fact with the exception of the `RandomForest` algorithm, the remaining ones are able to learn the classifiers in less than 1 s.

Confirming the trend related to the supervised classification algorithms, also considering the unsupervised classification algorithms, we obtain an increment of the performance. As a matter of fact, the value of precision and recall are increased for both the features categories we considered in the unsupervised machine learning:

Table 6. Feature Selection Classification results: Precision, Recall, F-Measure and RocArea for classifying the feature resulting of the feature selection process with the features of PI and PA categories, computed with six different classification algorithms [44].

Category	Algorithm	Precision	Recall	F-Measure	Roc Area	Time
<i>PI_selected</i>	J48	0.954	0.984	0.969	0.824	0.36
	DecisionStump	0.954	0.984	0.969	0.823	0.05
	HoeffdingTree	0.953	0.985	0.968	0.845	0.41
	RandomForest	0.943	0.944	0.943	0.831	15.06
	RandomTree	0.943	0.944	0.943	0.774	0.37
	REPTree	0.954	0.984	0.969	0.837	0.33
<i>PA_selected</i>	J48	0.958	0.986	0.972	0.870	0.48
	DecisionStump	0.957	0.971	0.964	0.830	0.15
	HoeffdingTree	0.957	0.985	0.971	0.882	0.25
	RandomForest	0.960	0.986	0.973	0.890	56.67
	RandomTree	0.954	0.954	0.954	0.818	0.98
	REPTree	0.959	0.985	0.972	0.889	0.74

Table 7. Feature Selection Classification results: Precision, Recall, F-Measure and RocArea for classifying the feature resulting of the feature selection process with the features of PI and PA categories, computed with six different unsupervised classification algorithms.

Category	Algorithm	Precision	Recall	F-Measure	Roc Area	Time
<i>PI_selected</i>	SimpleKMeans	0.97	0.95	0.95	0.896	0.08
	FarthestFirst	0.99	0.87	0.92	0.867	0.12
	FilteredClusterer	0.97	0.95	0.95	0.891	0.14
	MakeDensityBasedClusterer	0.97	0.95	0.95	0.891	12.04
	Cobweb	0.98	0.84	0.93	0.864	11.27
	DBScan	0.97	0.86	0.93	0.859	12.58
<i>PA_selected</i>	SimpleKMeans	0.96	0.94	0.94	0.848	0.12
	FarthestFirst	0.96	0.94	0.94	0.838	0.24
	FilteredClusterer	0.94	0.92	0.91	0.831	0.18
	MakeDensityBasedClusterer	0.97	0.95	0.95	0.891	12.04
	Cobweb	0.97	0.83	0.88	0.827	14.86
	DBScan	0.96	0.85	0.09	0.839	15.02

- relating to the PI category, the best precision value it is incremented from 0.934 to 0.95;
- relating to the PA category, the best precision value it is incremented from 0.954 to 0.97.

With regards to the time performance analysis, we confirm an increasing number in the time to learn the several unsupervised classifiers.

6 Conclusions

Online games are a widespread form of entertainment allowing users geographically distributed to play together, obtain upgrades for their characters by defeating monsters and enemies, and earn the game currency that can be changed in the real money. In this scenario, same attackers can use game bots to acquire a higher number of game points since game bots are able to play without any interruption disturbing the game of the human players. In this paper the method introduced in [10, 11] is described and largely evaluated. The aim is to discriminate an human user from a game bot classifying a set of behavioral features. We consider a set of features related to the player and to the game. On this features a classification is performed by obtaining in the best case a precision equal to 0.96 and a recall equal to 0.986. Moreover the same evaluation is also repeated by using cluster analysis algorithms and the best obtained results are good (0.99 for the best precision and 0.95 for the best recall). As future work we plan to investigate whether the feature vector we considered in this work is able to detect game bots in social network. Furthermore, we will consider the adoption of Process Mining techniques [45] and Formal Methods [46] in order to extract the game bots patterns with the aim to verify whether are different from the human users one.

Acknowledgements. This work has been partially supported by H2020 EU-funded projects NeCS and C3ISP and EIT-Digital Project HII.

References

1. Adams, E.: Fundamentals of Game Design (2014)
2. Quandt, T., Kröger, S.: Multiplayer: The Social Aspects of Digital Gaming, vol. 3. Routledge, Abingdon (2013)
3. Seay, A.F., Jerome, W.J., Lee, K.S., Kraut, R.E.: Project massive: a study of online gaming communities. In: CHI 2004 Extended Abstracts on Human Factors in Computing Systems, pp. 1421–1424. ACM (2004)
4. Yee, N.: Maps of digital desires: exploring the topography of gender and play in online games. In: Beyond Barbie and Mortal Kombat: New Perspectives on Gender and Gaming, pp. 83–89 (2008)
5. Griffiths, M.D., Davies, M.N., Chappell, D.: Online computer gaming: a comparison of adolescent and adult gamers. *J. Adolesc.* **27**, 87–96 (2004)
6. Chen, Y.C., Chen, P.S., Song, R., Korba, L.: Online gaming crime and security issue-cases and countermeasures from Taiwan. In: PST, pp. 131–136 (2004)
7. Paulson, R.A., Weber, J.E.: Cyberextortion: an overview of distributed denial of service attacks against online gaming companies. *Issues Inf. Syst.* **7**, 52–56 (2006)
8. Yampolskiy, R.V., Govindaraju, V.: Embedded noninteractive continuous bot detection. *Comput. Entertain. (CIE)* **5**, 7 (2008)
9. Kang, A.R., Jeong, S.H., Mohaisen, A., Kim, H.K.: Multimodal game bot detection using user behavioral characteristics. *SpringerPlus* **5**, 523 (2016)
10. Cabello, E., Cardoso, J., Ludwig, A., Maciaszek, L.A., van Sinderen, M. (eds.): ICISOFT 2016. CCIS, vol. 743. Springer, Cham (2017). <https://doi.org/10.1007/978-3-319-62569-0>

11. Bernardi, M.L., Cimitile, M., Mercaldo, F.: A time series classification approach to game bot detection. In: *Proceeding of the 7th ACM International Conference on Web Intelligence, Mining and Semantics*, pp. 512–519 (2017)
12. Varvello, M., Voelker, G.M.: Second life: a social network of humans and bots. In: *Proceedings of the 20th International Workshop on Network and Operating Systems Support for Digital Audio and Video, NOSSDAV 2010*, pp. 9–14. ACM, New York (2010)
13. Oh, J., Borbora, Z.H., Sharma, D., Srivastava, J.: Bot detection based on social interactions in MMORPGs. In: *2013 International Conference on Social Computing (SocialCom)*, pp. 536–543. IEEE (2013)
14. Kang, A.R., Woo, J., Park, J., Kim, H.K.: Online game bot detection based on party-play log analysis. *Comput. Math. Appl.* **65**, 1384–1395 (2013)
15. Kim, H., Hong, S., Kim, J.: Detection of auto programs for MMORPGs. In: Zhang, S., Jarvis, R. (eds.) *AI 2005. LNCS (LNAI)*, vol. 3809, pp. 1281–1284. Springer, Heidelberg (2005). https://doi.org/10.1007/11589990_187
16. Chen, K.T., Pao, H.K.K., Chang, H.C.: Game bot identification based on manifold learning. In: *Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games*, pp. 21–26. ACM (2008)
17. Chen, K.-T., Liao, A., Pao, H.-K.K., Chu, H.-H.: Game bot detection based on avatar trajectory. In: Stevens, S.M., Saldamarco, S.J. (eds.) *ICEC 2008. LNCS*, vol. 5309, pp. 94–105. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89222-9_11
18. Chen, K.T., Jiang, J.W., Huang, P., Chu, H.H., Lei, C.L., Chen, W.C.: Identifying MMORPG bots: a traffic analysis approach. In: *Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology, ACE 2006*. ACM, New York (2006)
19. Kwon, H., Mohaisen, A., Woo, J., Kim, Y., Lee, E., Kim, H.K.: Crime scene reconstruction: online gold farming network analysis. *IEEE Trans. Inf. Forensics Secur.* **12**, 544–556 (2017)
20. Kim, H., Yang, S., Kim, H.K.: Crime scene re-investigation: a postmortem analysis of game account stealers' behaviors. *CoRR abs/1705.00242* (2017)
21. Thawonmas, R., Kashifuji, Y., Chen, K.T.: Detection of MMORPG bots based on behavior analysis. In: *Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, pp. 91–94. ACM (2008)
22. Kashifuji, Y.: Detection of MMORPG bots based on behavior analysis. In: *ACE 2008* (2008)
23. Hilaire, S., Kim, H., Kim, C.: How to deal with bot scum in MMORPGs? In: *2010 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR)*, pp. 1–6. IEEE (2010)
24. Mishima, Y., Fukuda, K., Esaki, H.: An analysis of players and bots behaviors in MMORPG. In: *2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)*, pp. 870–876. IEEE (2013)
25. Chung, Y., Park, C.Y., Kim, N., Cho, H., Yoon, T.B., Lee, H., Lee, J.: A behavior analysis-based game bot detection approach considering various play styles. *CoRR abs/1509.02458* (2015)
26. Mitchell, T.M.: Machine learning and data mining. *Commun. ACM* **42**, 30–36 (1999)

27. Michalski, R.S., Carbonell, J.G., Mitchell, T.M.: Machine Learning: An Artificial Intelligence Approach. Springer, Heidelberg (2013). <https://doi.org/10.1007/978-3-662-12405-5>
28. Estai, M., Kanagasingam, Y., Xiao, D., Vignarajan, J., Bunt, S., Kruger, E., Tennant, M.: End-user acceptance of a cloud-based teledentistry system and Android phone app for remote screening for oral diseases. *J. Telemed. Telecare* **23**, 44–52 (2017)
29. Canfora, G., De Lorenzo, A., Medvet, E., Mercaldo, F., Visaggio, C.A.: Effectiveness of opcode ngrams for detection of multi family Android malware. In: 2015 10th International Conference on Availability, Reliability and Security (ARES), pp. 333–340. IEEE (2015)
30. Canfora, G., Mercaldo, F., Visaggio, C.A.: A classifier of malicious Android applications. In: 2013 Eighth International Conference on Availability, Reliability and Security (ARES), pp. 607–614. IEEE (2013)
31. Ling, C.X., Yang, Q., Wang, J., Zhang, S.: Decision trees with minimal costs. In: Proceedings of the Twenty-First International Conference on Machine learning, p. 69. ACM (2004)
32. Jin, C., De-Lin, L., Fen-Xiang, M.: An improved ID3 decision tree algorithm. In: 4th International Conference on Computer Science and Education, ICCSE 2009, pp. 127–130. IEEE (2009)
33. Pang, J., Huang, Q., Jiang, S.: Multiple instance boost using graph embedding based decision stump for pedestrian detection. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5305, pp. 541–552. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88693-8_40
34. Hang, Y., Fong, S.: Investigating the impact of bursty traffic on Hoeffding Tree Algorithm in stream mining over internet. In: 2010 Second International Conference on Evolving Internet (INTERNET), pp. 147–152. IEEE (2010)
35. Liaw, A., Wiener, M., et al.: Classification and regression by randomForest. *R News* **2**, 18–22 (2002)
36. Cutler, A., Zhao, G.: Pert-perfect random tree ensembles. *Comput. Sci. Stat.* **33**, 490–497 (2001)
37. Zhao, Y., Zhang, Y.: Comparison of decision tree methods for finding active objects. *Adv. Space Res.* **41**, 1955–1959 (2008)
38. Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C., Silverman, R., Wu, A.Y.: The analysis of a simple k-means clustering algorithm. In: Proceedings of the Sixteenth Annual Symposium on Computational Geometry, pp. 100–109. ACM (2000)
39. Kumar, M., et al.: An optimized farthest first clustering algorithm. In: 2013 Nirma University International Conference on Engineering (NUiCONE), pp. 1–5. IEEE (2013)
40. Panda, M., Patra, M.: A novel classification via clustering method for anomaly based network intrusion detection system. *Int. J. Recent Trends Eng.* **2**, 1–6 (2009)
41. Pandey, A.K., Pandey, P., Jaiswal, K., Sen, A.K.: Datamining clustering techniques in the prediction of heart disease using attribute selection method. *Heart Dis.* **14**, 16–17 (2013)
42. Fisher, D.H.: Knowledge acquisition via incremental conceptual clustering. *Mach. Learn.* **2**, 139–172 (1987)
43. Dua, S., Du, X.: Data Mining and Machine Learning in Cybersecurity. CRC Press, Boca Raton (2016)

44. Bernardi, M.L., Cimitile, M., Distante, D., Mercaldo, F.: Game bot detection in online role player game through behavioural features. In: Proceeding of the 12th International Conference on Software Technologies (2017)
45. Bernardi, M.L., Cimitile, M., Di Francescomarino, C., Maggi, F.M.: Do activity lifecycles affect the validity of a business rule in a business process? *Inf. Syst.* **62**, 42–59 (2016)
46. Francesco, N.D., Lettieri, G., Santone, A., Vaglini, G.: Heuristic search for equivalence checking. *Softw. Syst. Model.* **15**, 513–530 (2016)